

MC514–Sistemas Operacionais: Teoria e Prática
1s2009

Gerenciamento de Memória - 2

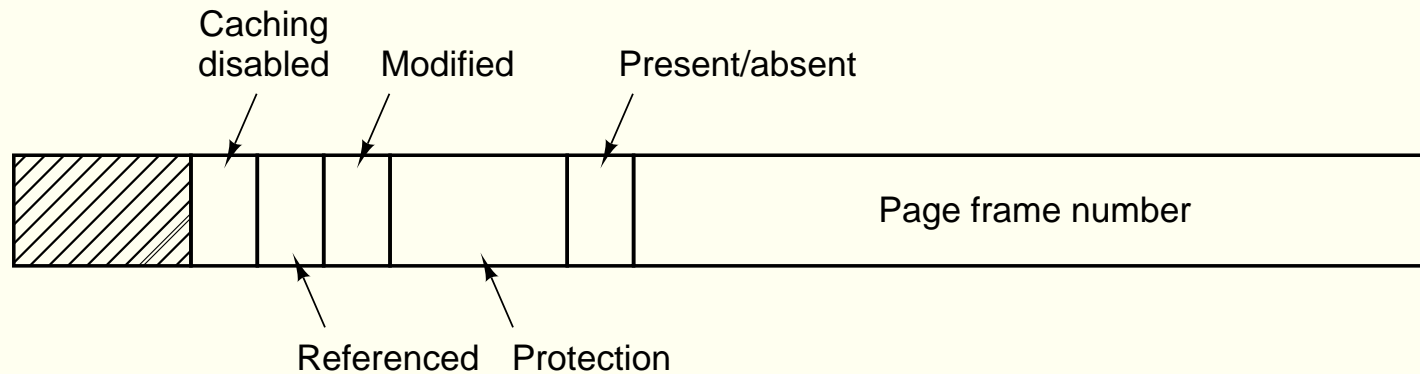
Substituição de páginas

- Veja os códigos: pag1.c pag2.c

Algoritmo ótimo:

- Baseado no uso futuro de uma página
- Impossível de ser implementado
- Pode ser simulado (segunda execução do mesmo processo com a mesma entrada)
- Útil para medidas de desempenho

Entrada na tabela de páginas



Tanenbaum: Figura 4.13

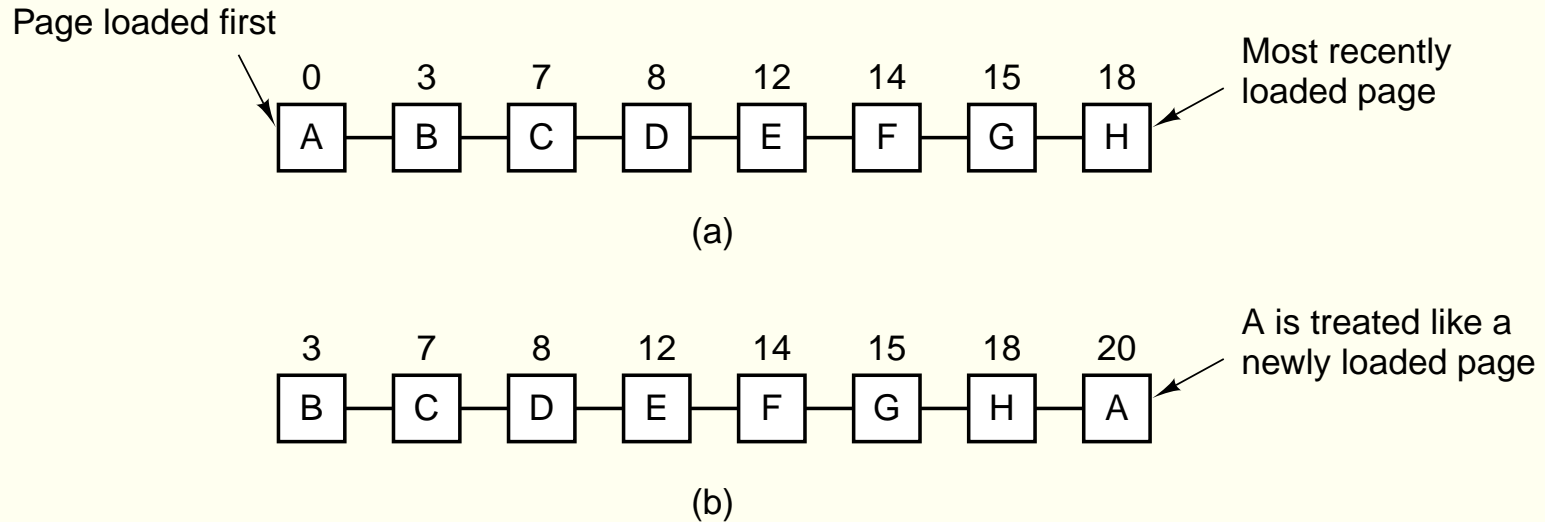
Não usada recentemente

- Classe 0: não referenciada, não modificada
- Classe 1: não referenciada, mas modificada
- Classe 2: referenciada, mas não modificada
- Classe 3: referenciada e modificada

First In, First Out

- Coloca as páginas em uma fila
- Pode remover páginas importantes

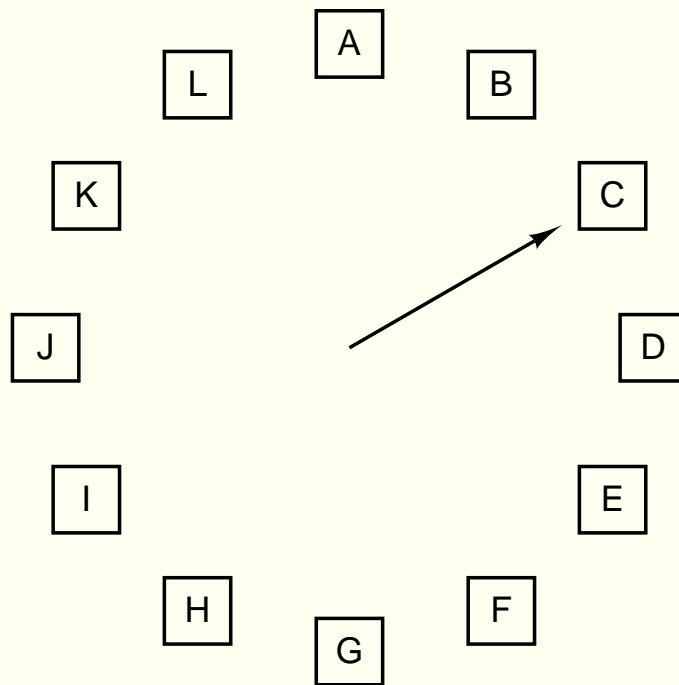
Segunda chance



Tanenbaum: Figura 4.16

- Se o bit $R == 0$, a página é substituída, senão
- bit R é limpo e a página é colocada no final da fila

Relógio



When a page fault occurs,
the page the hand is
pointing to is inspected.
The action taken depends
on the R bit:

R = 0: Evict the page

R = 1: Clear R and advance hand

Tanenbaum: Figura 4.17

- Implementação circular da segunda chance

Uso menos recente

- LRU (Least Recently Used)
- Implementação utilizando lista ligada
- Implementação em hardware com contador
 - incrementado a cada instrução
 - entrada na tabela deve armazenar o contador
- Implementação com matriz $n \times n$

LRU em hardware

	Page			
	0	1	2	3
0	0	1	1	1
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0

(a)

	Page			
	0	1	2	3
0	0	0	1	1
1	1	0	1	1
2	0	0	0	0
3	0	0	0	0

(b)

	Page			
	0	1	2	3
0	0	0	0	1
1	1	0	0	1
2	1	1	0	1
3	0	0	0	0

(c)

	Page			
	0	1	2	3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0

(d)

	Page			
	0	1	2	3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	1
3	1	1	0	0

(e)

0	0	0	0
1	0	1	1
1	0	0	1
1	0	0	0

(f)

0	1	1	1
0	0	1	1
0	0	0	1
0	0	0	0

(g)

0	1	1	0
0	0	1	0
0	0	0	0
1	1	1	0

(h)

0	1	0	0
0	0	0	0
1	1	0	1
1	1	0	0

(i)

0	1	0	0
0	0	0	0
1	1	0	0
1	1	1	0

(j)

Acessos: 0 1 2 3 2 1 0 3 2 3

Tanenbaum: Figura 4.18

Simulando LRU em software

Uso não frequente

- Contador de uso para cada página
(soma o bit R a cada clock tick)
- Não esquece nada...
- Considere um compilador baseado em passos

Uso não frequente

Aging

- O contador é deslocado à direita
- Bit R é adicionado à esquerda

Página referenciada

1 0 1 0 1 1

>> 1 1 0 1 0 1

+ 1 1 1 0 1 0 1

Página não referenciada

1 0 1 1 0 1

>> 1 1 0 1 1 0

+ 0 0 1 0 1 1 0

- Quais são as limitações desta abordagem?

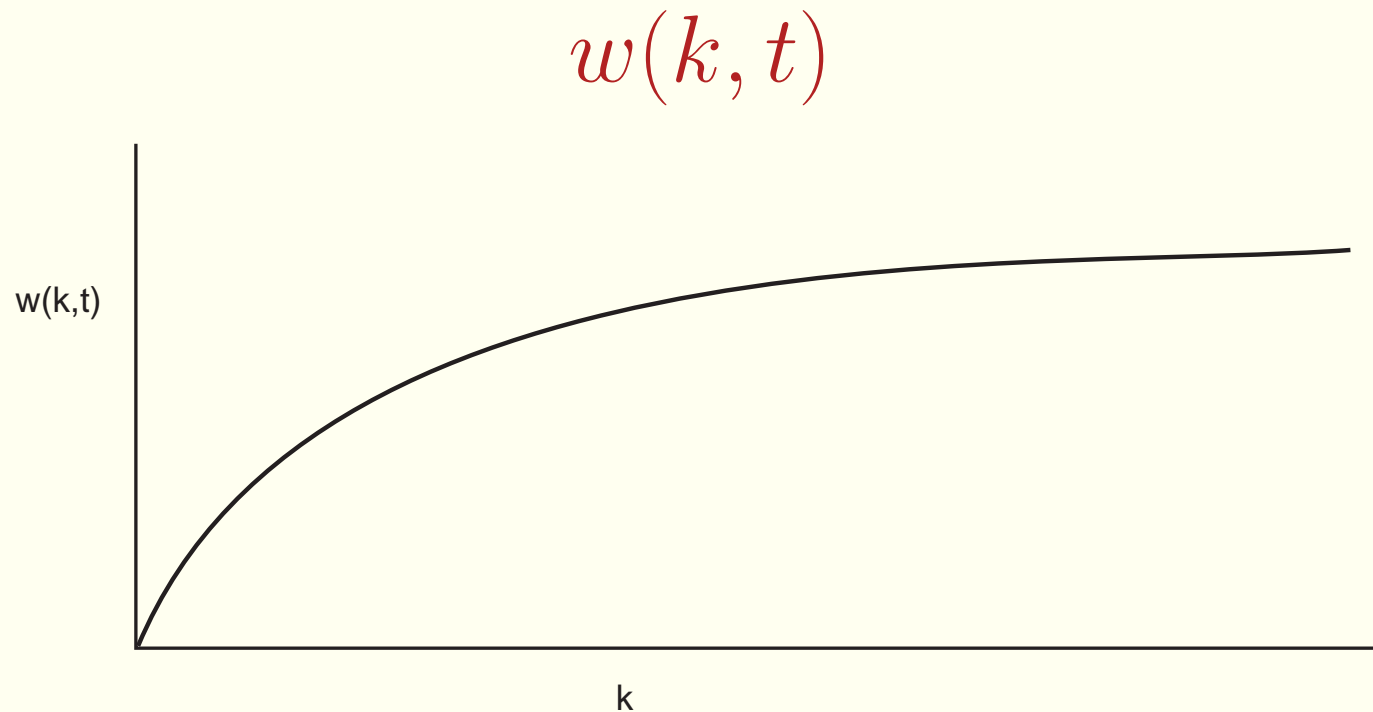
Uso não frequente

	R bits for pages 0-5, clock tick 0	R bits for pages 0-5, clock tick 1	R bits for pages 0-5, clock tick 2	R bits for pages 0-5, clock tick 3	R bits for pages 0-5, clock tick 4
	1 0 1 0 1 1	1 1 0 0 1 0	1 1 0 1 0 1	1 0 0 0 1 0	0 1 1 0 0 0
Page					
0	10000000	11000000	11100000	11110000	01111000
1	00000000	10000000	11000000	01100000	10110000
2	10000000	01000000	00100000	00100000	10001000
3	00000000	00000000	10000000	01000000	00100000
4	10000000	11000000	01100000	10110000	01011000
5	10000000	01000000	10100000	01010000	00101000
	(a)	(b)	(c)	(d)	(e)

Tanenbaum: Figura 4.19

Working Set

- Paginação sob demanda
- Localidade de referência
- Prepaging
- Como implementar?



Tanenbaum: Figura 4.20

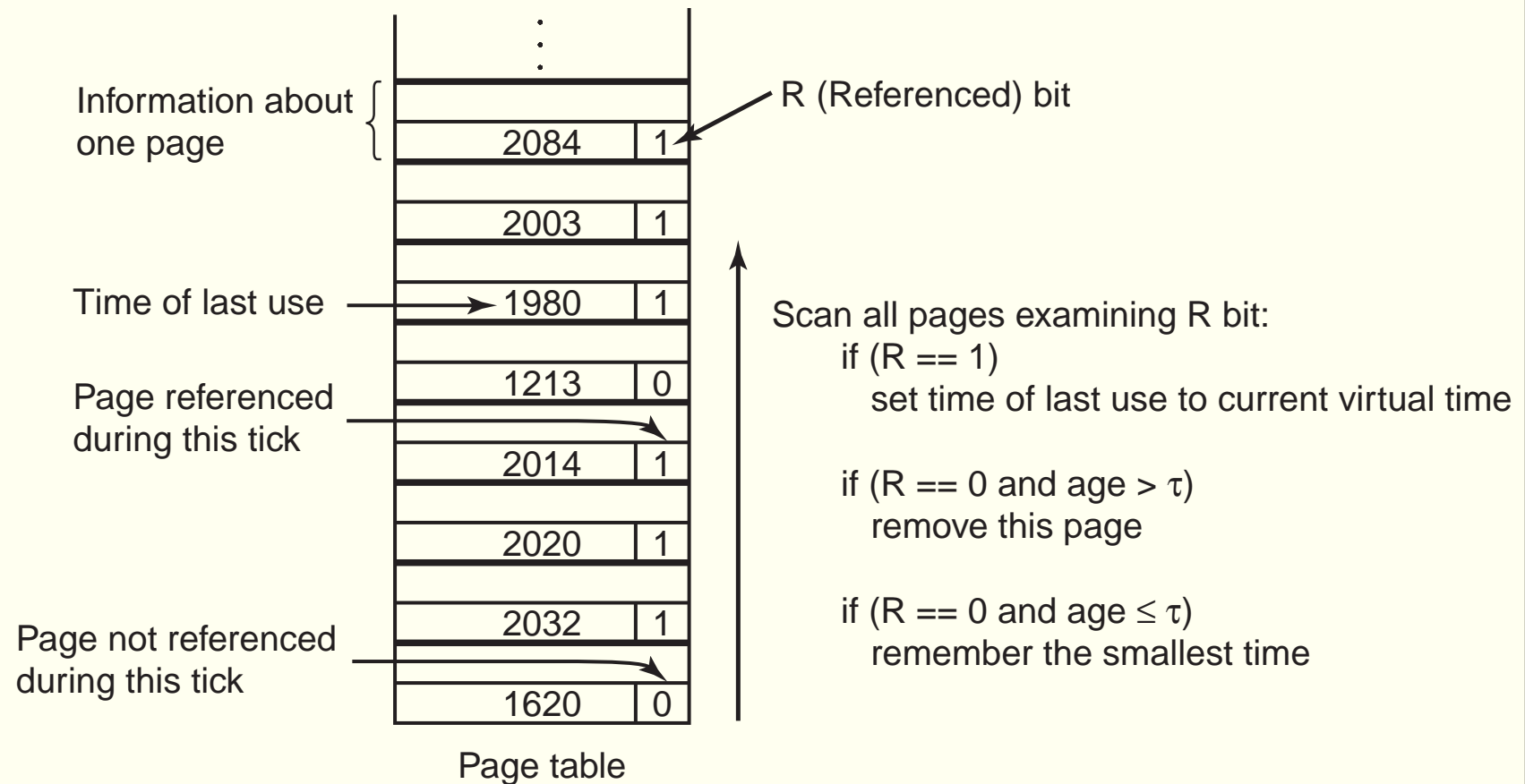
- $w(k, t)$: número de páginas utilizadas nas últimas k referências em relação ao instante t .

Working Set

- Últimas k referências a memória
- Referências no último δ intervalo de tempo
 - tempo virtual

Working Set

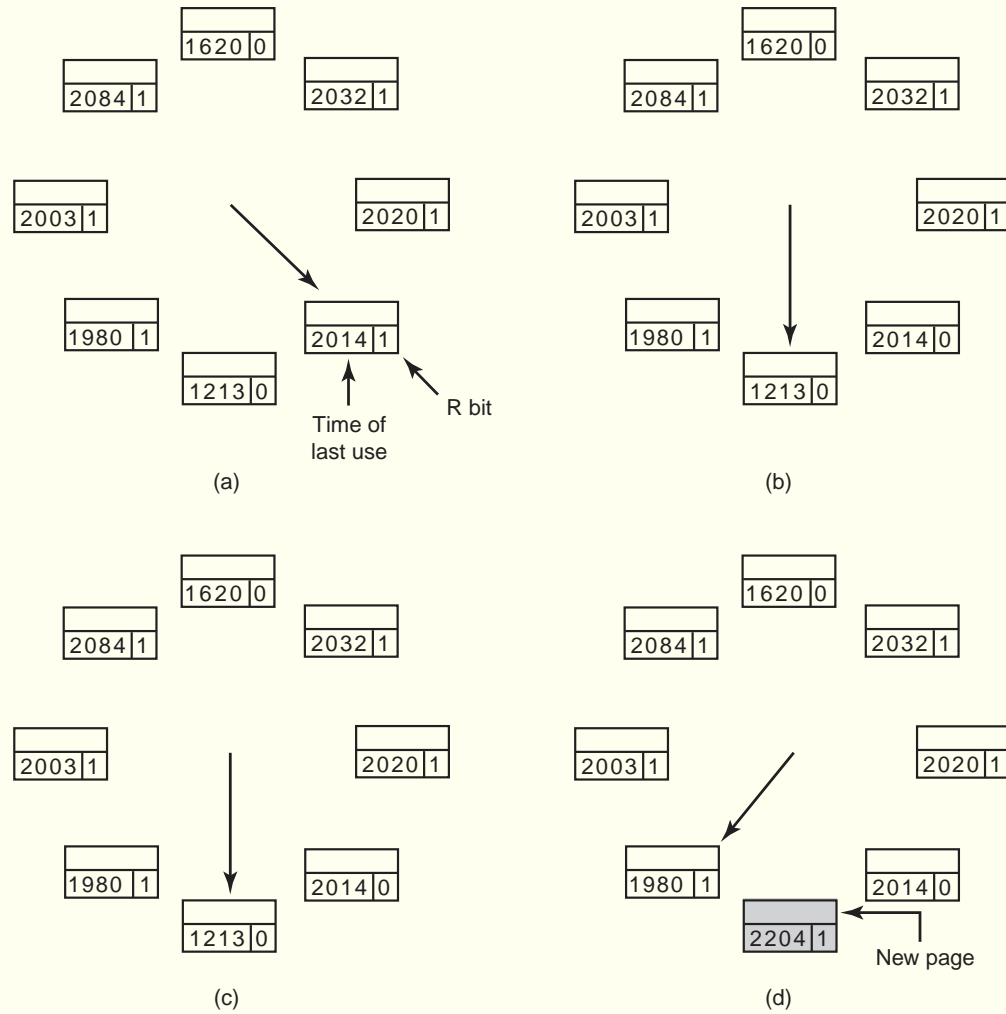
2204 Current virtual time



Tanenbaum: Figura 4.21

WSClock

2204 Current virtual time



Resumo

Algorithm	Comment
Optimal	Not implementable, but useful as a benchmark
NRU (Not Recently Used)	Very crude
FIFO (First-In, First-Out)	Might throw out important pages
Second chance	Big improvement over FIFO
Clock	Realistic
LRU (Least Recently Used)	Excellent, but difficult to implement exactly
NFU (Not Frequently Used)	Fairly crude approximation to LRU
Aging	Efficient algorithm that approximates LRU well
Working set	Somewhat expensive to implement
WSClock	Good efficient algorithm

Tanenbaum: Figura 4.23

Política de limpeza

- Gravar as páginas modificadas na última hora pode ser pouco eficiente
- *Paging Daemon*
 - Varre periodicamente a memória
 - Tenta manter um número de *frames* livres
 - kswapd

Controle da carga

- *Thrashing*
 - O Sistema Operacional só se ocupa das tarefas de paginação e escalonamento
 - Todos os processos precisam de mais memória
 - Swap (como escolher quais processos vão para o disco?)

Fixação de páginas na memória

- Melhora do desempenho para processos com características especiais

```
int mlock (const void *ADDR, size_t LEN);  
int munlock (const void *ADDR, size_t LEN);
```

- Só para super-usuários? :-)

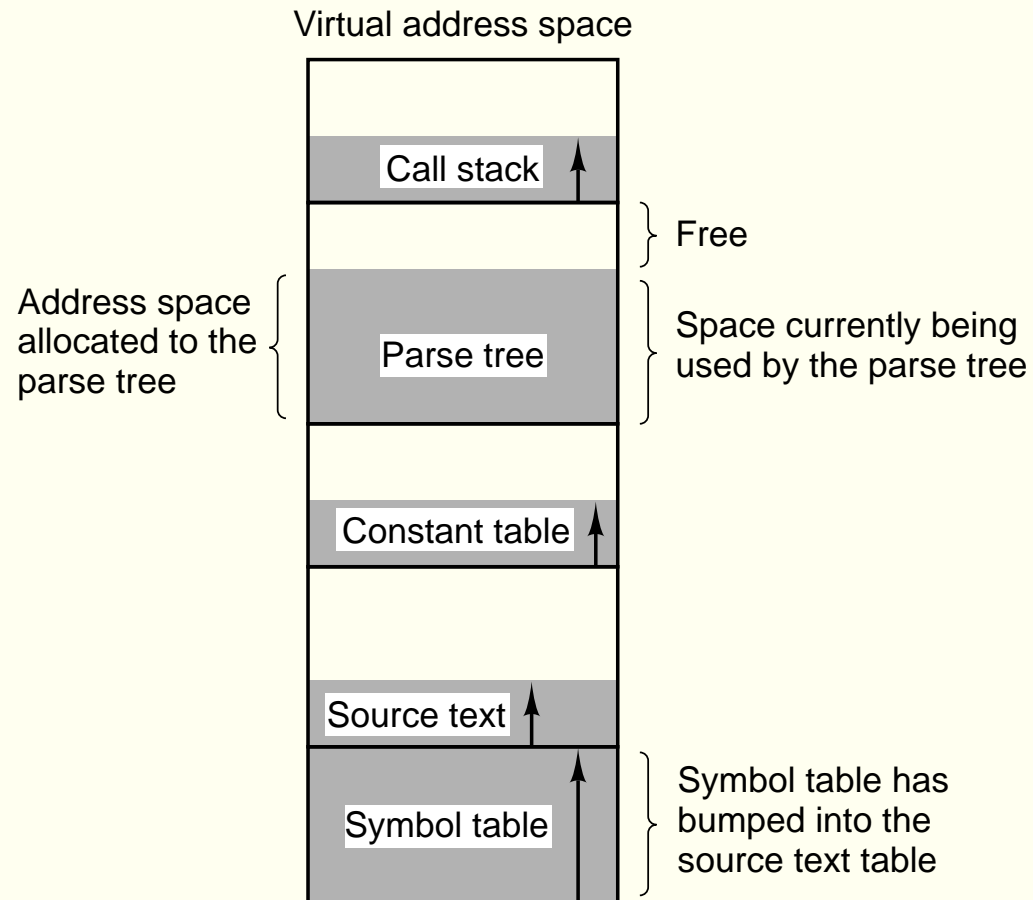
Mmap

- Um arquivo pode ser mapeado em memória
- Memória compartilhada
- Veja map.c

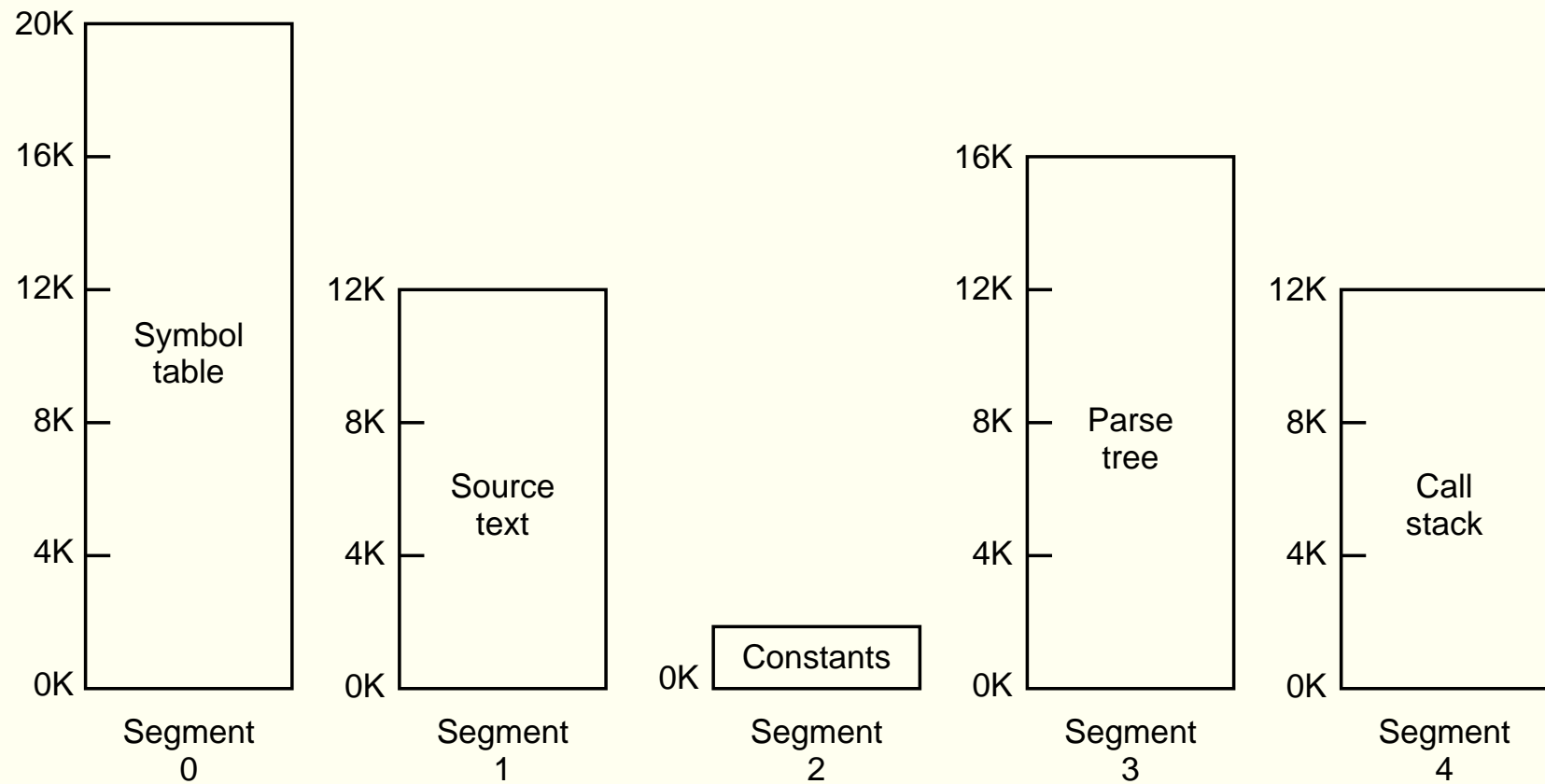
Segmentação

- Vários espaços de endereçamento
- Maior flexibilidade

Limites de um espaço único

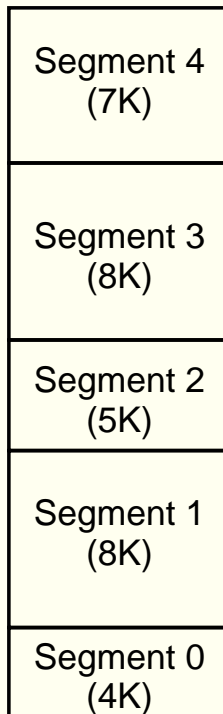


Segmentação

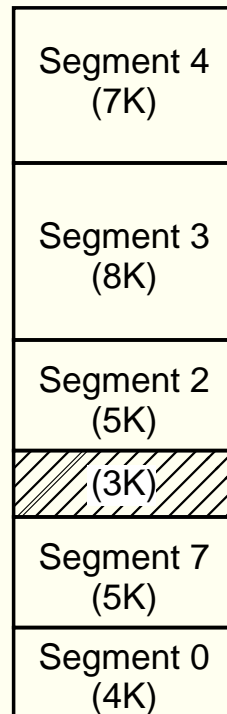


Consideration	Paging	Segmentation
Need the programmer be aware that this technique is being used?	No	Yes
How many linear address spaces are there?	1	Many
Can the total address space exceed the size of physical memory?	Yes	Yes
Can procedures and data be distinguished and separately protected?	No	Yes
Can tables whose size fluctuates be accommodated easily?	No	Yes
Is sharing of procedures between users facilitated?	No	Yes
Why was this technique invented?	To get a large linear address space without having to buy more physical memory	To allow programs and data to be broken up into logically independent address spaces and to aid sharing and protection

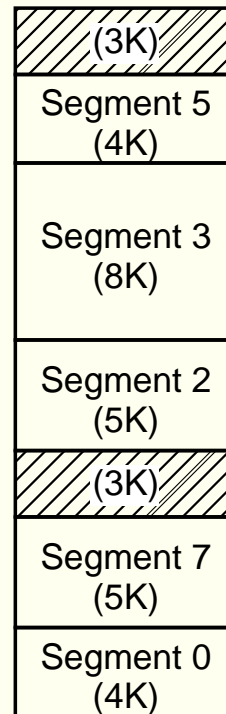
Segmentação pura



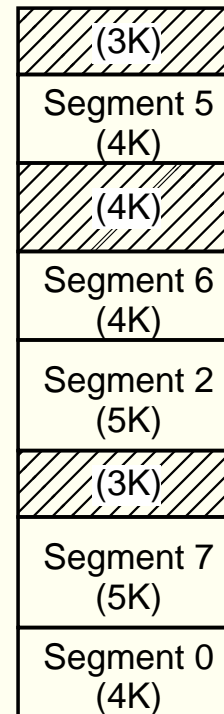
(a)



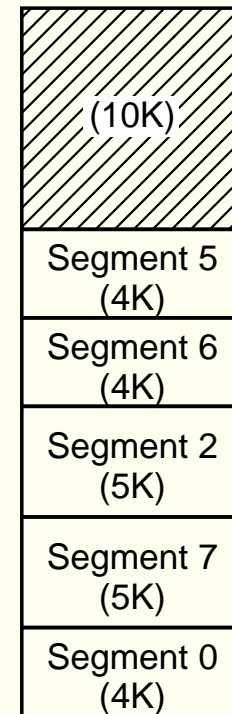
(b)



(c)

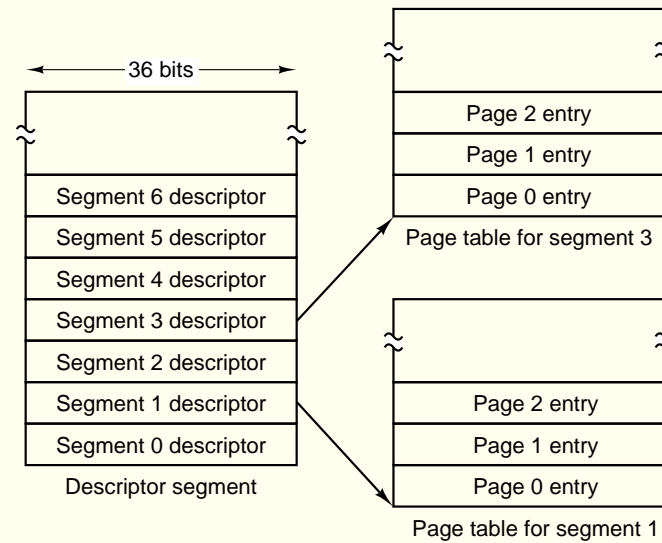


(d)

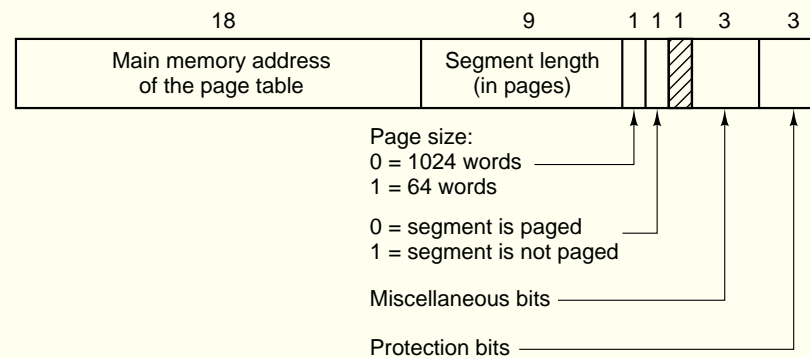


(e)

MULTICS: segmentação e paginação



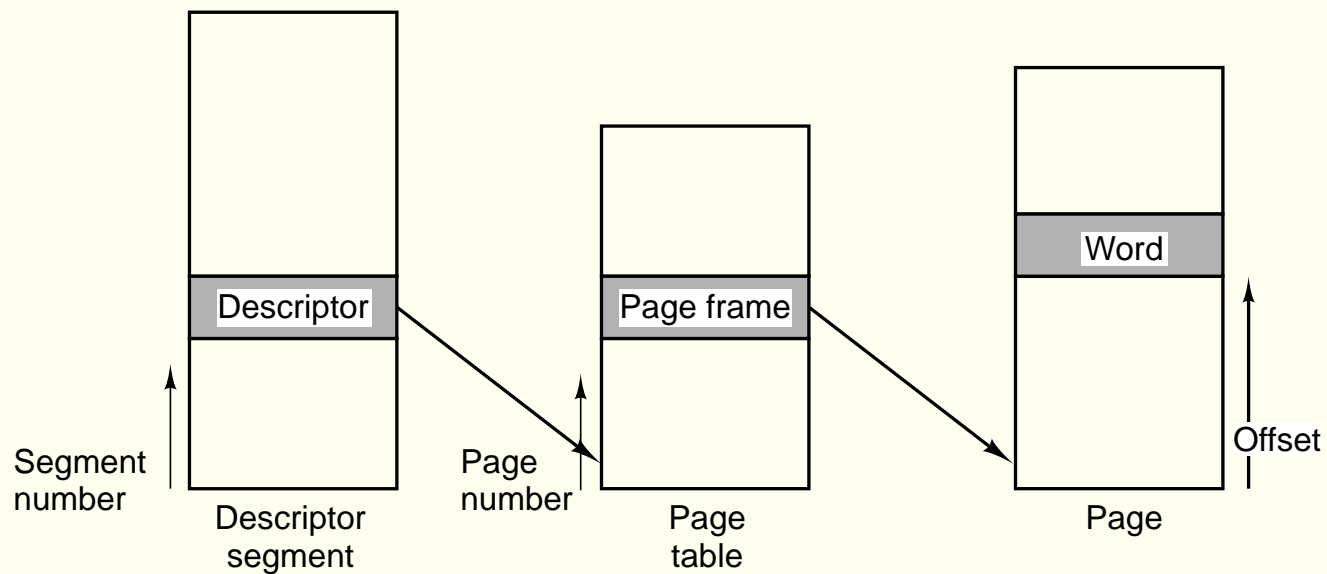
(a)



(b)

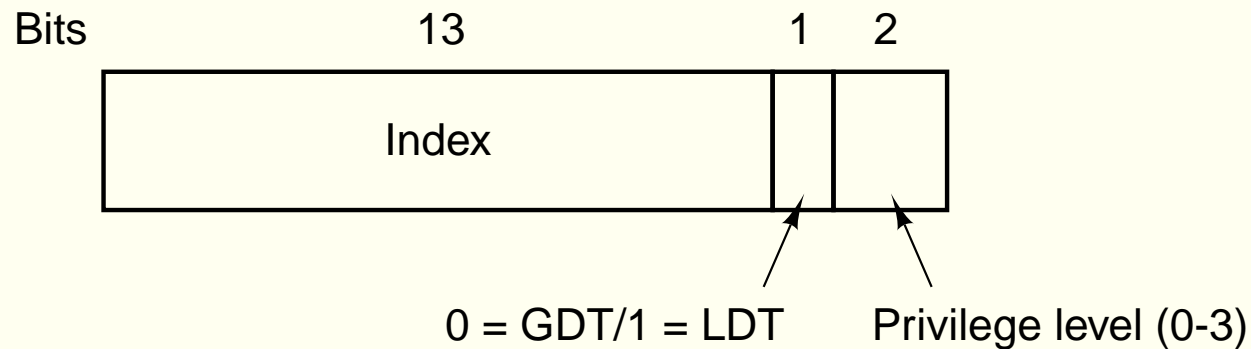
MULTICS: Conversão de endereços

MULTICS virtual address



Intel Pentium: segmentação e paginação

- Seletor de segmento

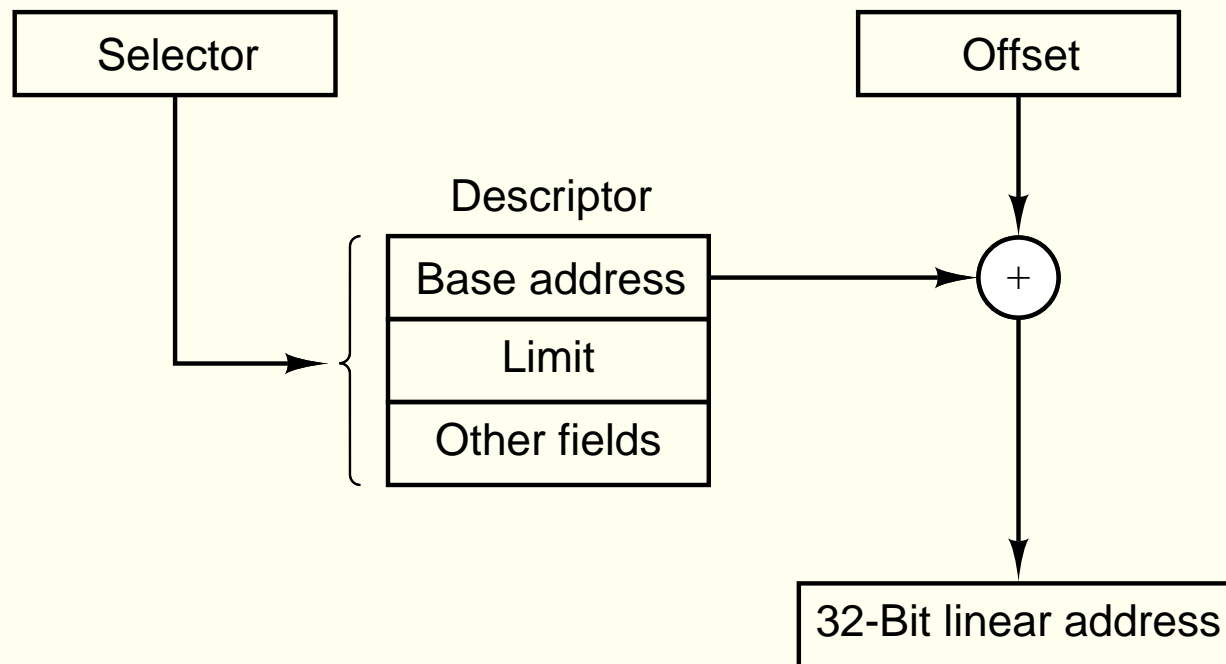


- LDT (Local Descriptor Table)
- GDT (Global Descriptor Table)

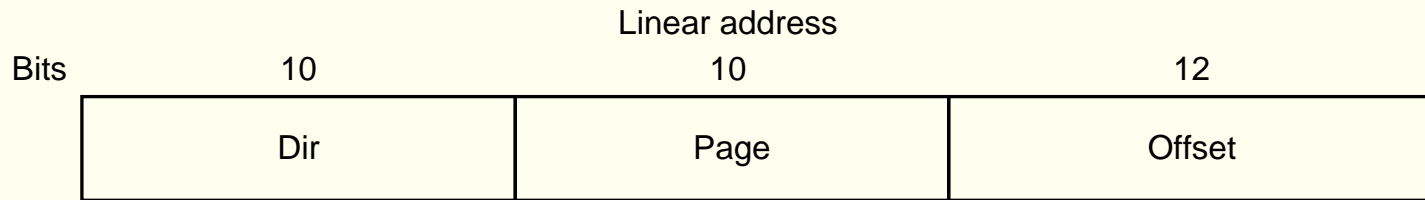
Registradores de segmento

- CS - Code Segment
- DS - Data Segment
- SS - Stack Segment
- ES, FS, GS - Extra Segments

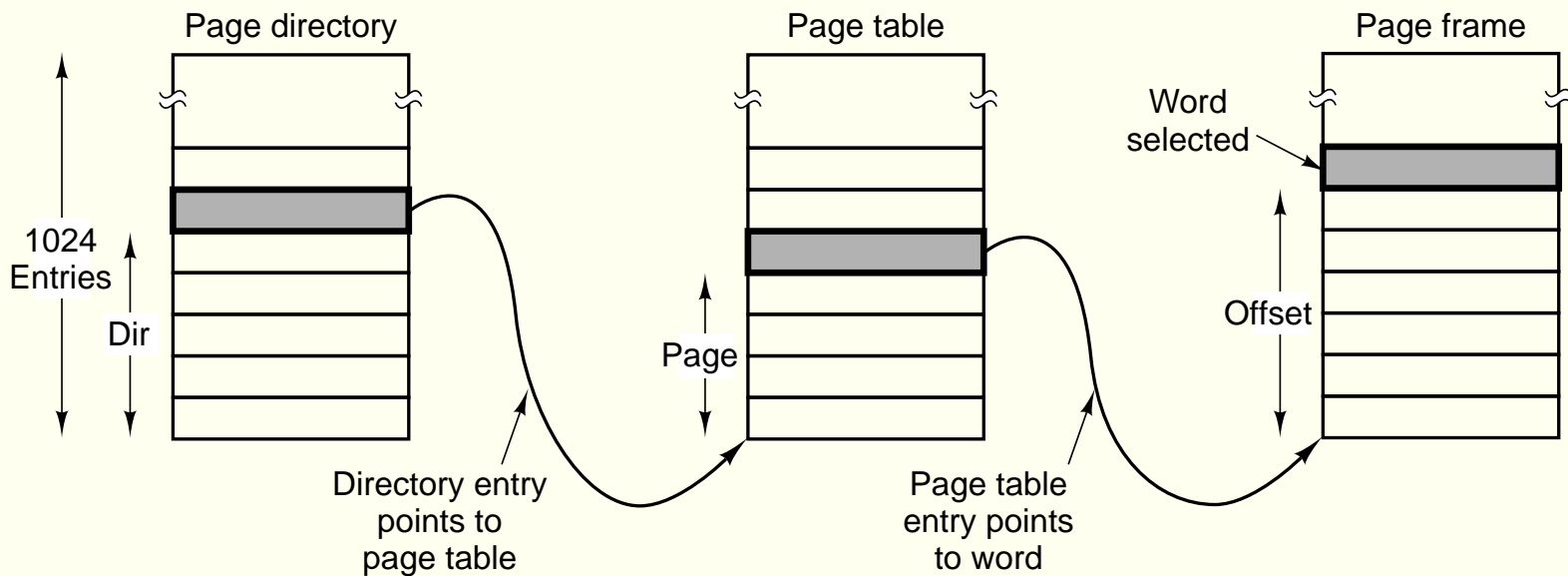
Conversão de endereços (seletor, deslocamento) \rightarrow linear



Conversão de endereços linear \rightarrow físico



(a)



(b)